

The Hong Kong Polytechnic University
Department of Computing

COMP4913 Capstone Project
Final Report

RF-based indoor localization system

Student Name:	CHEUK Lok Kan
Student ID No.:	19053031D
Programme-stream:	BSc (Hons) in Information Technology (61431-FIT)
Supervisor:	Dr YANG Ray
Submission Date:	6 th April 2023

Abstract

Precision-demanding localization is a key research field that much expertise pay effort in order to enhance its reliability and coverage. The topic can be divided as outdoor and indoor. Outdoor localization is no longer an issue nowadays due to the ongoing development of global positioning system. However, when it comes to indoor localization, there are many inherent and irresistible issues causing it to perform far from expected. It is no wonder that hardware and spatial diversity, multipath effect due to obstacles, and mathematical errors by coordinate conversion are the main culprits of the inaccurate performance (ACM, 2022). In hopes of enhancing the accuracy of the indoor localization prediction in order that it can be applied in practical scenarios contextually requiring the accuracy rather than coverage. Considering the result to be achieved, we reduce the indoor localization prediction problem into a mathematical joint non-linear optimization problem so that deep learning can be taken part in boosting the accuracy of indoor localization prediction (DataFlair, n.d.). To ensure the quality and quantity of datasets for deep learning, a tailor-made database, named “Ray”, consisting of a million-scale dataset with millimeter-level labels is adopted as the source of data being free of protocol for the training of the neural network. Also, “ThreeBodyNet”, an innovative deep neural network (DNN) is proposed for indoor localization to showcase the potentials of the designated database. One of the scenes that “ThreeBodyNet” acquires data from the “Ray” database is the beginning of the analysis of the indoor localization prediction.

Table of contents

Abstract	2
1 Background and Related Work	4
2 Objectives and Outcomes	8
3 Design and Methodology	15
4 Implementation and Results.....	22
5 Evaluation	25
6 Conclusion	28
7 Resources Utilized	29
8 References.....	30

1 Background and Related Work

Nowadays, there are measures tackling the unavailability of GPS service when it comes to indoor localization for the last hundred meters such as using radio-frequency identification (RFID), Bluetooth, and wireless fidelity (SmartX, n.d.). Despite facilitating various key applications within buildings such as indoor navigation, location-based advertising, and Internet of Things (IoT) management, indoor localization is rarely deployed in reality because of a variety of errors across deployment scenes. For example, diversities in hardware and electromagnetic fields, influenced signal propagation due to obstacles by means of shadowing, reflection, scattering, and diffraction, and mathematical mistakes during coordinate conversion. Hardware diversity from circuitry characteristics causes errors of radio-frequency (RF) signals, introducing unknown noises to the input and therefore reducing the quality of results. Multipath effects reflect RF signals from different obstacles in uncertain and complex indoor wireless environments by hindering the line-of-sight (LoS) propagation. Coordinate conversion is interfered when converting intermediate measurements into a result in the global coordinate system because of the distributivity of the localization system that deploying several gateways in different positions. Such complexity demands a high-precision system to identify the gateways before integrating them into the entire system (Jin, 2006). The last two reasons causing errors are remarkable that they restrain its applications in practical scenarios requiring rigorous accuracy such as searching for a particular book out of dozens of books placed on a large shelf in a library.

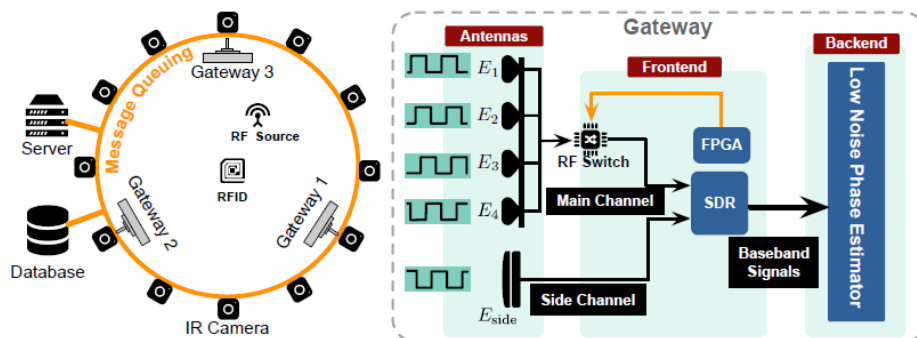


Figure 1: Deployment and architecture of the gateway

Env. (#)	Scene (#)	Setting (#)	RSS (dBm)	1G (#)	2G (#)	3G (#)	Total (#)	Density (p/m ³)	Space (m ²)	MP. (#)	Dist. (m)	Size (GB)	DR. (r/s)	TS. (min)	Temp. (°C)
Semi-Indoor Env.	A	S1	-62.5	32,191	35,308	16,893	84,392	3,843.0	78.5	10	5	48.24	40.4	79.5	31.2
		S2	-66.4	21,551	24,480	11,300	57,311	4,689.9	314.2	8	10	28.47	38.0	44.4	30.3
		S3	-66.7	20,372	23,773	11,382	55,527	5,274.2	706.9	7	15	26.88	42.4	41.2	29.9
		S4	-69.4	19,483	24,858	10,177	54,518	3,787.0	1,256.6	9	20	24.18	41.0	35.1	29.4
		S5	-71.0	16,567	22,278	11,457	50,302	4,336.4	1,963.5	10	25	25.38	40.1	35.0	27.2
		S6	-75.0	18,414	21,818	11,009	51,241	5,865.5	2,827.4	9	30	22.68	38.2	32.9	27.4
		S7	-77.4	16,445	22,807	12,037	51,289	5,871.0	3,848.5	10	35	23.82	35.9	33.8	27.7
		S8	-78.8	28,672	35,540	10,309	74,521	7,834.4	5,026.5	10	40	35.28	40.5	50.8	28.1
		S9	-79.3	28,672	35,540	10,309	61,909	4,235.7	6,361.7	8	45	27.66	41.4	40.9	28.3
		S10	-79.1	32,919	33,753	9,803	76,475	5,223.7	7,854.0	13	50	25.80	41.2	63.6	29.0
		S11	-88.6	29,440	17,387	3,359	50,186	10,490.4	7,854.0	11	55	18.15	31.5	29.8	28.7
	B	S12	-71.8	4,301	10,683	8,044	23,028	538.7	1,963.5	10	25	43.62	6.2	69.5	30.1
		S13	-76.9	6,245	10,172	4,940	21,357	702.1	3,848.5	13	35	39.07	5.9	66.9	30.4
		S14	-78.1	6,942	19,579	11,782	38,303	1,381.8	5,026.5	12	40	64.44	6.8	113.1	30.9
	C	S15	-68.3	5,533	9,075	4,118	18,726	6,079.9	1,256.6	16	20	40.02	7	74.5	33.1
	D	S16	-68.9	24,007	32,448	21,083	77,538	4,345.1	530.9	7	13	21.48	33.9	39.5	29.2
		S17	-67.0	15,684	16,584	8,303	40,571	2,545.9	530.9	10	13	46.92	32.2	81.6	28.8
Full-Indoor Env.	E	S18	-66.2	2,326	31,693	126,475	160,494	38,212.9	314.2	14	10	89.93	20.7	61.2	18.4
		S19	-65.3	8,720	69,915	N/A	78,635	27,924.4	314.2	11	10	31.25	27.4	124.1	24.9
		S20	-63.7	3,173	26,930	N/A	30,103	10,906.9	314.2	11	10	12.19	16.1	26.5	25.1
		S21	-64.9	2,998	23,918	N/A	26,916	8,900.8	314.2	10	10	11.74	39.3	21.7	27.6
		S22	-65.4	18,872	13,170	N/A	32,042	5,057.1	314.2	11	10	12.47	40.7	18.0	24.8
	F	S23	-65.1	4,930	17,714	25,823	48,467	22,627.0	153.9	9	7	71.46	8.1	124.1	25.8
	G	S24	-61.4	1,749	4,222	4,550	10,521	4,911.8	78.5	8	5	11.88	4.2	65.9	27.5
		S25	-60.2	891	2,425	1,975	5,291	937.8	78.5	5	5	9.42	4.3	33.6	30.1
		S26	-61.9	1,911	3,973	839	6,723	2,394.2	78.5	12	5	11.94	5.3	31.0	27.3
		S27	-61.9	1,593	3,527	3,293	8,413	1,828.9	78.5	10	5	17.28	4.1	100.1	28.2
		S28	-63.7	1,297	3,526	4,088	8,911	1,600.4	78.5	6	5	13.92	9.6	57.5	27.9
		S29	-61.7	1,026	1,984	2,092	5,102	912.7	113.1	7	6	14.11	3.7	33.1	29.3
	H	S30	-60.4	526	1,335	1,789	3,650	1,011.6	113.1	5	6	9.72	5.1	17.8	28.8
		S31	-60.9	964	2,555	3,947	7,466	823.0	113.1	7	6	16.02	8.9	28.2	29.9
	I	S32	-61.0	528	1,244	982	2,754	307.9	113.1	9	6	12.72	5.4	17.2	28.0
		S33	-61.6	9,379	16,164	N/A	25,543	1,576.7	78.5	8	5	8.41	48.3	11.8	18.5
	K	S34	-71.0	25,988	2,894	N/A	28,882	1,380.6	1256.6	4	20	11.78	29.6	16.6	17.8
	L	S35	-77.7	23,743	28,891	N/A	52,634	935.9	1963.5	10	25	23.85	27.8	34.4	17.6
	M	S36	-79.9	8,874	12,809	N/A	21,683	1,335.2	3217.0	8	32	8.53	24.0	12.3	17.3
	N	S37	-68.5	14,250	7,479	N/A	21,729	848.1	254.5	6	9	7.83	30.6	11.3	16.8

Table 1: Summary of “Ray” database

In order to overcome the errors being dependent on the deployment scenes and preventing further applications, the “Ray” database, consisting of over 1.32 million datasets with millimeter-level labels, is adopted as the source of data by customizing it with three 4×4 antenna arrays (Figure 1) and each antenna is at a distance of 16cm (Figure 2). For the data collected by RFID tags, they are protocol-free and general-purpose and so suitable to train the neural network (Zeng, 2019). Besides, the datasets in the database are from 14 different scenes, including full-indoor environments and semi-indoor environments, with 37 different settings (Figure 3&4). The largest coverage is up to 7,854m² (Table 1). Also, the precision of labels is not an issue by adopting OptiTrack, an optical positioning system capturing labels with precision in denomination of millimeter. As a result, the quality and quantity of data for indoor localization prediction by DNN is guaranteed.

Gathering the qualified data from the “Ray” database, we obtain parameters for a mathematical joint non-linear optimization problem, simplified from the current indoor localization prediction

problem. It can be addressed by “ThreeBodyNet”, a novel and dedicated DNN for indoor localization, exemplifying the outstanding performance in accuracy and stability compared with current localization algorithms. The DNN contains three body networks, each of which is made of a ResNet-50 based convolutional neural network (Fox, 2004). Resultantly, “ThreeBodyNet” managed to achieve a median error of 0.127m, outperforming by 53% than recent localization neural networks.

To sum up, RF signals at three positions are collected by three distributed gateways from the customized “Ray” database at first. Then, a DNN for indoor localization prediction is proposed for a satisfying accuracy.

2 Objectives and Outcomes

Data analysis of the performance of RFID by means of DNN helps the localization of RFID tags. Affecting the accuracy of indoor localization prediction using RFID, Received Signal Strength Identification (RSSI) and phase values that each gateway receives from different positions of the RFID tag are important parameters for the data analysis. RSSI refers to the strength of RF signal received by gateways from the tag. Decreasing with increasing distance, RSSI is useful to determine the distance between the gateways and the tag by detecting the signal frequency. However, inaccuracies in distance estimation due to environmental interference hinder the effectiveness of RSSI. Therefore, phase values are also taken into account for the accuracy of the localization prediction by representing the phase difference between the signal received by the gateways and that of transmitted by the tag, depending on the time delay between the transmission and reception of the signal. Higher phase values, greater phase distance and therefore lower signal quality due to multipath propagation and synchronization errors and vice versa. Consequently, phase values not only indicate the reliability of RSSI by constructive interference that the phase values of the received signal is comparable to that of reference signal and vice versa, but also by pointing out the obstructions during signal propagation, causing imprecisions in the localization prediction. This is because RSSI-based localization prediction assumes that the route that the signal travels is free of reflections and phase shifts. Predicting the coordinates of the indoor localization system holistically, we are to adopt both RSSI and phase values as the parameters of the data analysis by “ThreeBodyNet”, the proposed DNN (Shen, 2019).

Model1.pth	29/12/2022 0:33	PTH 檔案	216 KB
Model2.pth	29/12/2022 0:42	PTH 檔案	216 KB
Model3.pth	29/12/2022 0:51	PTH 檔案	216 KB
Model4.pth	29/12/2022 8:53	PTH 檔案	228 KB
Model5.pth	29/12/2022 9:03	PTH 檔案	228 KB
Model6.pth	29/12/2022 9:11	PTH 檔案	228 KB
Model7.pth	29/12/2022 9:22	PTH 檔案	241 KB
Prediction1.xlsx	29/12/2022 0:33	Microsoft Excel 工作...	321 KB
Prediction2.xlsx	29/12/2022 0:43	Microsoft Excel 工作...	319 KB
Prediction3.xlsx	29/12/2022 0:52	Microsoft Excel 工作...	325 KB
Prediction4.xlsx	29/12/2022 8:53	Microsoft Excel 工作...	3 KB
Prediction5.xlsx	29/12/2022 9:03	Microsoft Excel 工作...	553 KB
Prediction6.xlsx	29/12/2022 9:11	Microsoft Excel 工作...	551 KB
Prediction7.xlsx	29/12/2022 9:23	Microsoft Excel 工作...	788 KB
project.py	29/12/2022 22:26	Python File	5 KB
project.pyproj	29/12/2022 11:52	Python Project	2 KB

Figure 5: Predictions produced by respective models in “project.py”

After solving the problems that how RSSI and phase values affect the accuracy of indoor localization prediction and the mystery about the relationship between phase values and RSSI, the problem about “ThreeBodyNet”, the proposed and dedicated DNN for indoor localization, comes that how to design the DNN so as to showcase the potentials of “Ray”, the designated database, to the greatest degree. The solution is to intensively expand the number of layers in the DNN and the neurons in each layer. After continuous training, the data model with least test loss is adopted to be the implementation of “ThreeBodyNet”, generating several models and sets of prediction (Figure 5) referring to different arrangement and number of gateways.

RFID Localization System

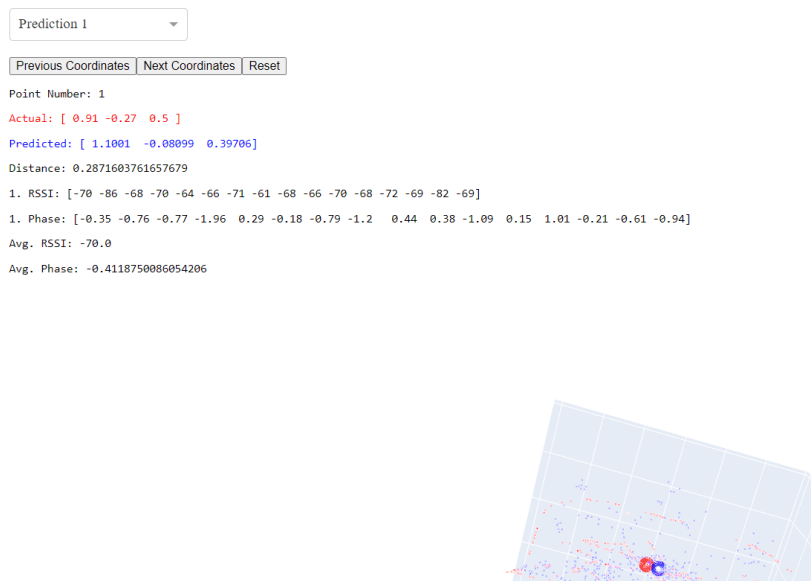


Figure 6: User interface of the scatter plot webpage

Gathering prediction sets of coordinates according to various scenarios, we face a problem that how to visualize pairs of actual and predicted coordinates to observe their distance since greater distance of coordinates means greater errors of models generated. A solution using a graphical user interface (GUI) in the form of webpage (Figure 6) by showing all pairs of actual and predicted coordinates in a 3D Cartesian coordinate system is selected. Only selected pair of coordinates are highlighted to show the corresponding predicted coordinates. Prediction sets in all scenarios are included in the webpage to exhibit each performance of prediction model using different number of gateways.

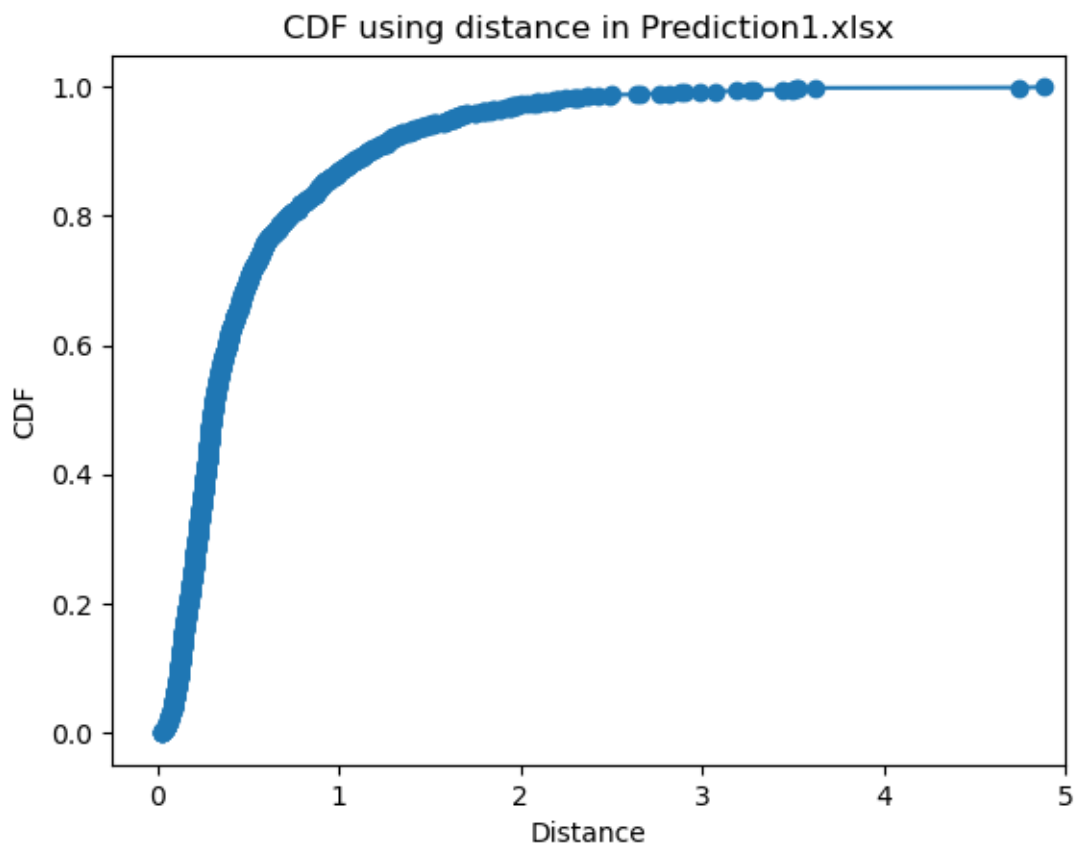


Figure 7: Cumulative Distribution Function (CDF) of distance between actual and predicted coordinates in Prediction 1

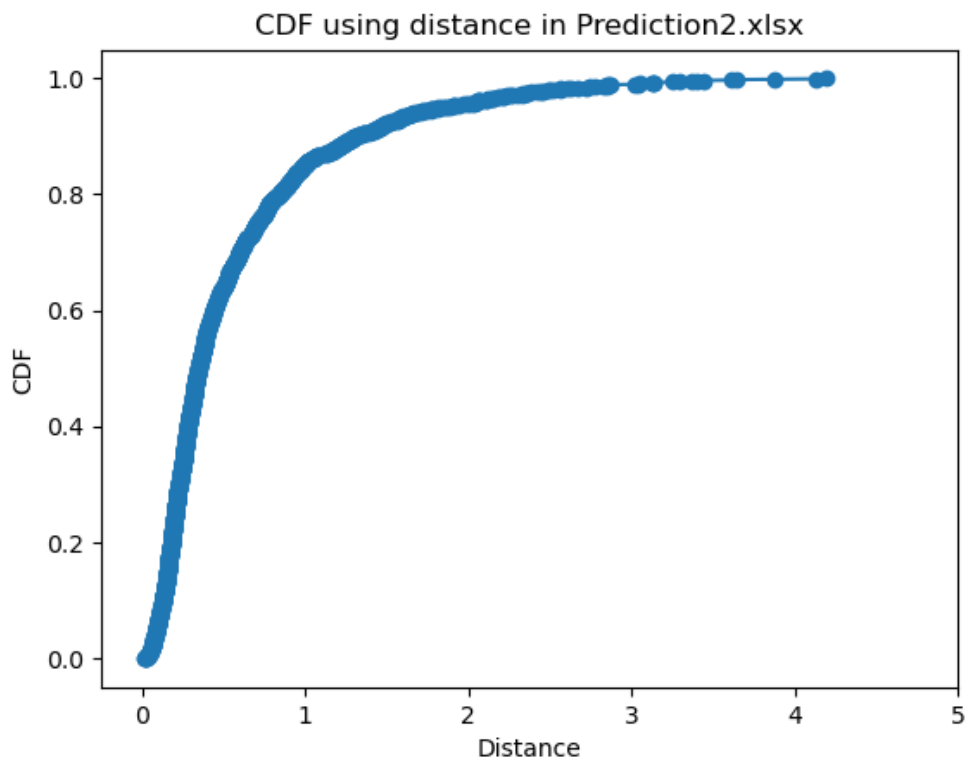


Figure 8: Cumulative Distribution Function (CDF) of distance between actual and predicted coordinates in Prediction 2

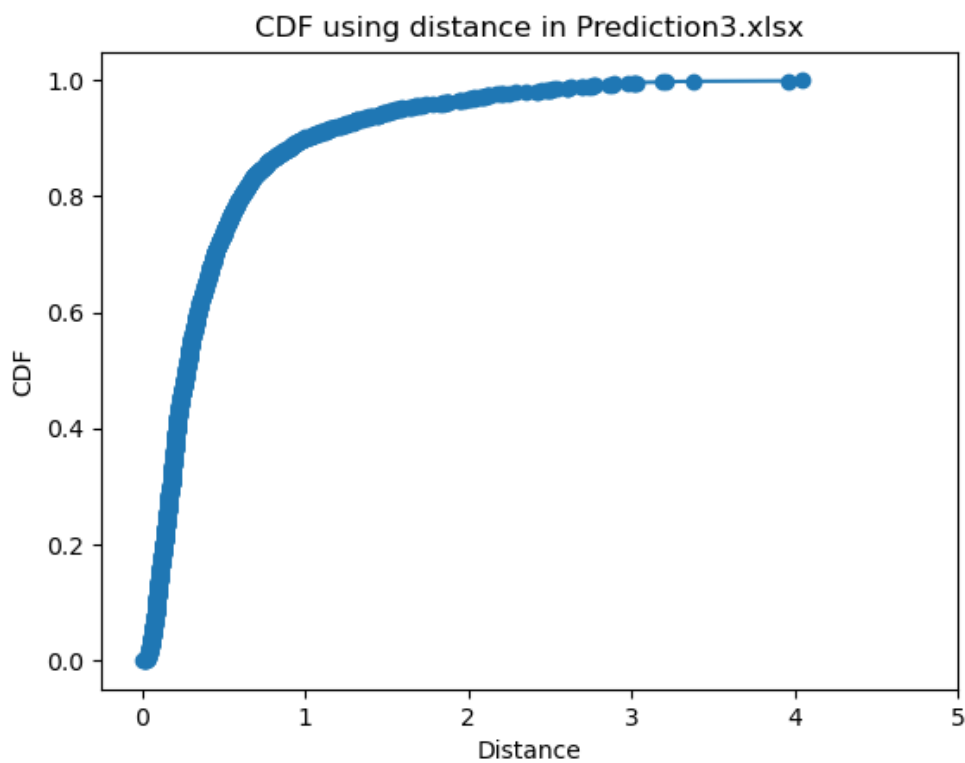


Figure 9: Cumulative Distribution Function (CDF) of distance between actual and predicted coordinates in Prediction 3

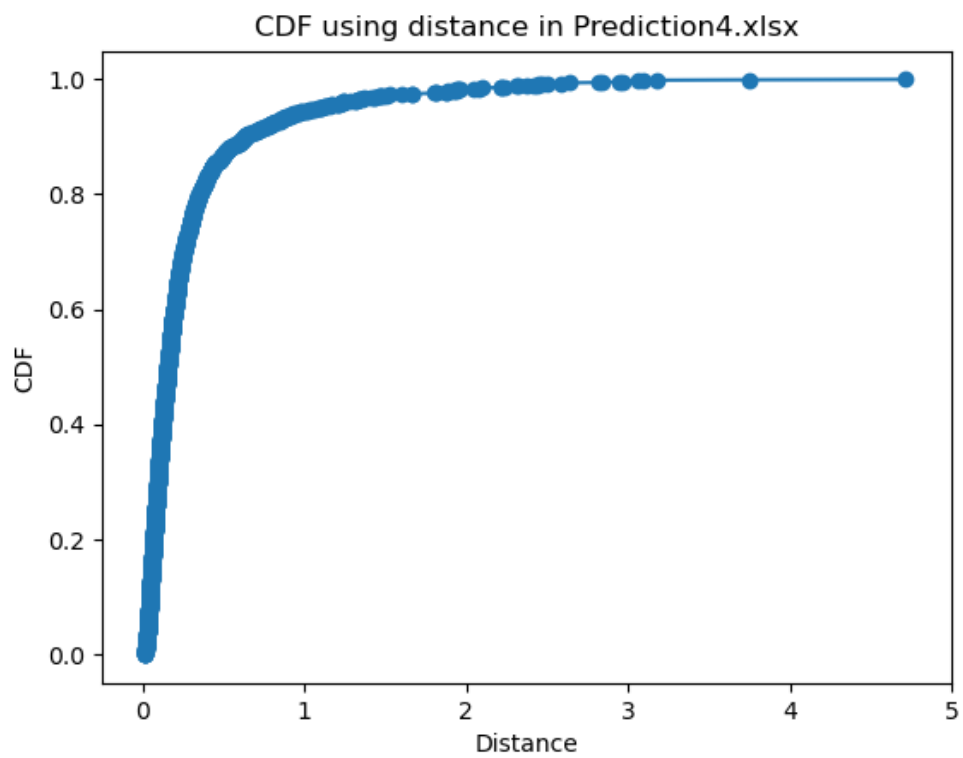


Figure 10: Cumulative Distribution Function (CDF) of distance between actual and predicted coordinates in Prediction 4

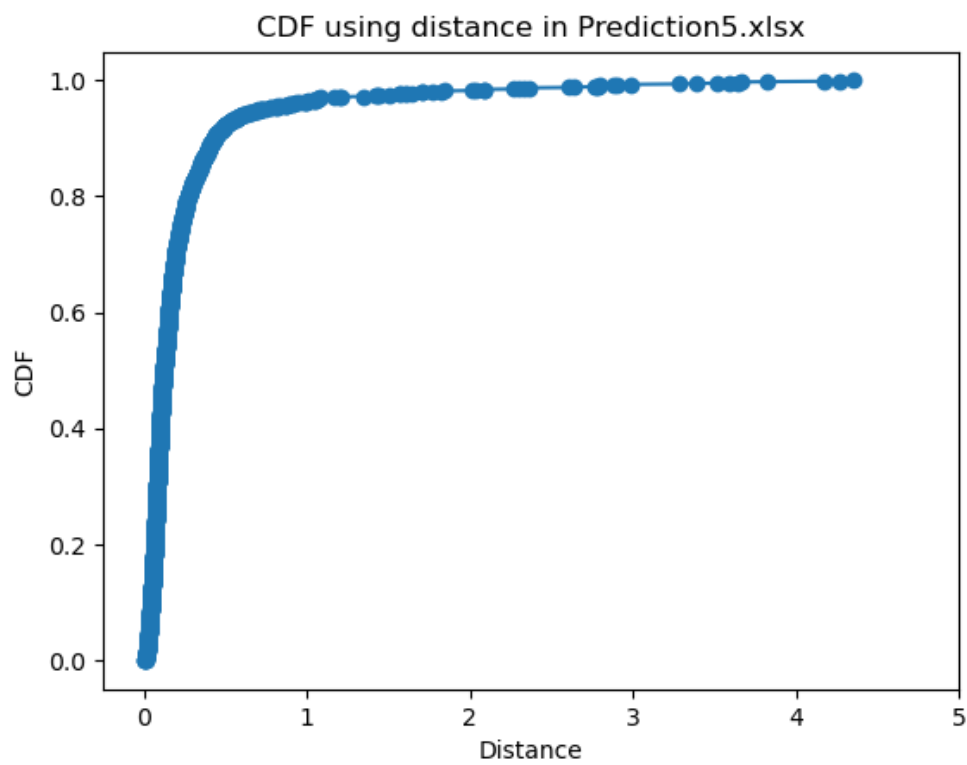


Figure 11: Cumulative Distribution Function (CDF) of distance between actual and predicted coordinates in Prediction 5

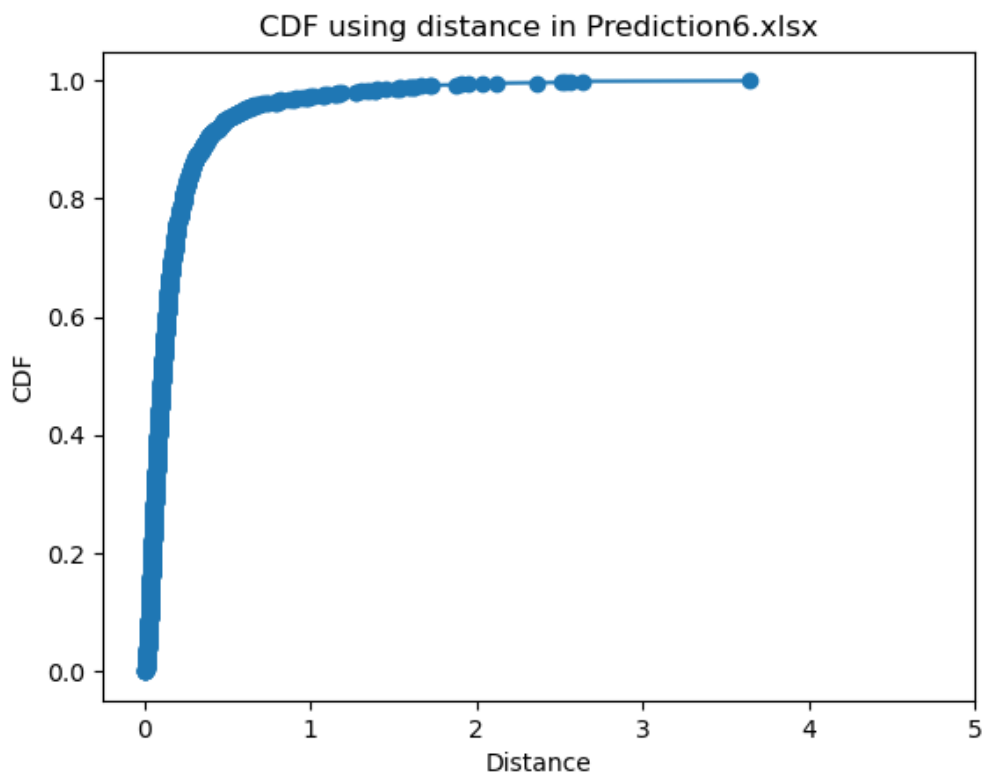


Figure 12: Cumulative Distribution Function (CDF) of distance between actual and predicted coordinates in Prediction 6

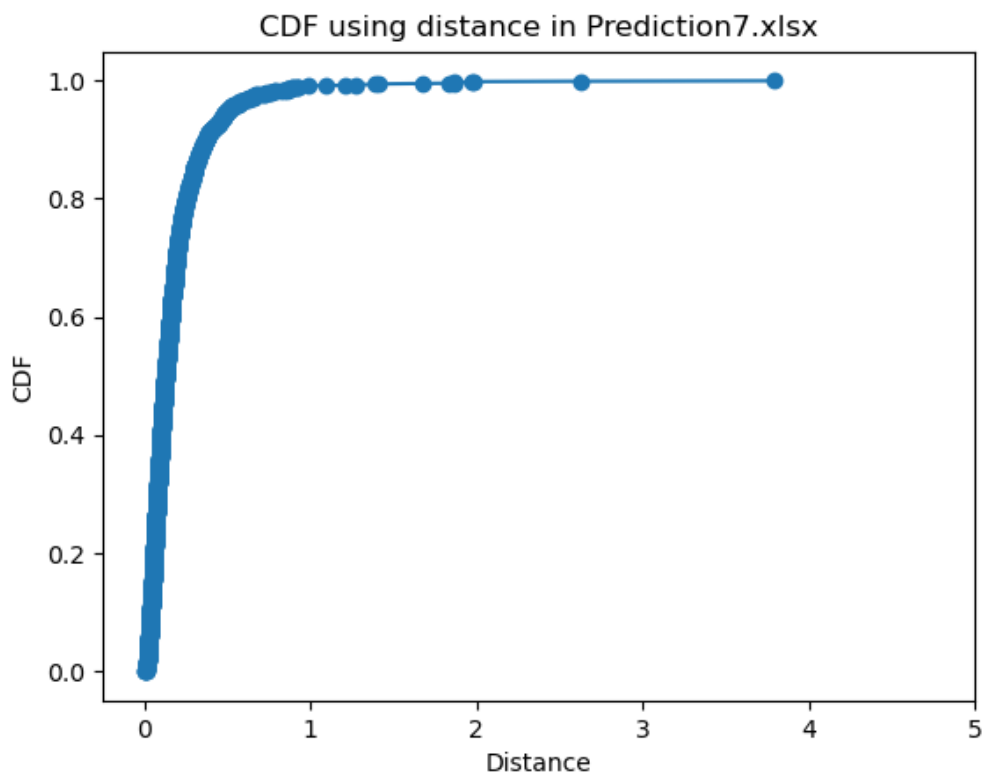


Figure 13: Cumulative Distribution Function (CDF) of distance between actual and predicted coordinates in Prediction 7

Visualizing all pairs of coordinates in a 3D Cartesian coordinate system, we finally face the last problem that how to mathematically indicate the distance of each pair of coordinates so that the holistic performance of each prediction model can be exemplified by revealing their extent of errors caused. A solution using cumulative distribution function (CDF) to find the cumulative distribution of distance between actual and predicted coordinates from every prediction set in order to monitor the performance of each prediction model is discovered. The more inclined between the intervals 0m and 1m (Figure 7-13), the higher precision of the prediction model by resulting more minor errors whereas fewer significant errors when predicting coordinates.

All in all, problems that how RSSI and phase values affect the accuracy of indoor localization prediction, relationship between phase values and RSSI, paradigm of designing “ThreeBodyNet” for generating the most accurate prediction models, visualization of pairs of actual and predicted coordinates for direct observation of their distance, and mathematically signify the distance of each pair of coordinates overall are the problems to be addressed and objectives to be achieved in this research. Eventually, all purposes are successfully accomplished to be the outcomes of the research.

3 Design and Methodology

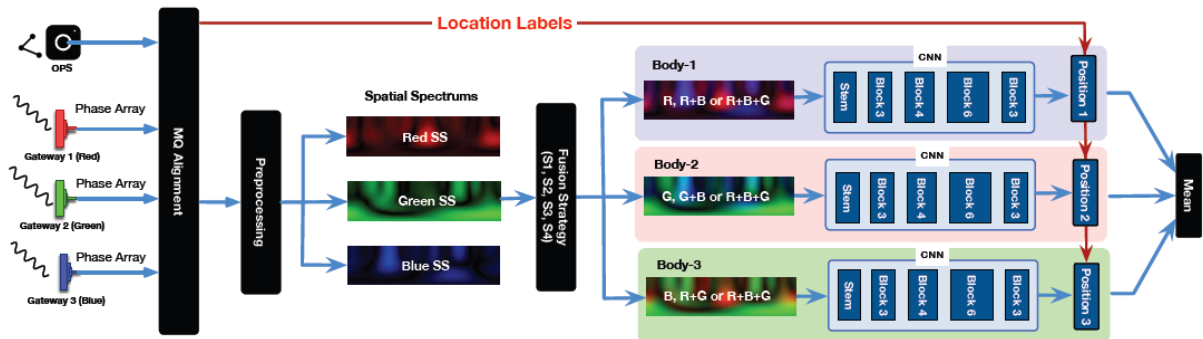


Figure 14: Architecture of “ThreeBodyNet”

	project.py	scatter.py	cdf.py
Python 3.10 with Miniconda3	✓	✓	✓
Conda 23.1.0	✓	✓	✓
NumPy 1.23.5	✓	✓	✓
PyTorch 1.12.1	✓		
pandas 1.5.3	✓		✓
Dash 2.7.0		✓	
Plotly 5.9.0		✓	
pickle 4.0		✓	
Matplotlib 3.7.1			✓
openpyxl 3.0.10			✓

Table 2: Technologies, frameworks, libraries, and packages used in each Python project

The project will be conducted by breaking into 3 small tasks, which are project.py, scatter.py, and cdf.py. In these tasks, Python and multifarious technologies are used (Table 2). For the first task project.py, it simulates the architecture of “ThreeBodyNet” (Figure 14) that gateway 1 represents red spatial spectrum, gateway 2 represents green spatial spectrum, and gateway 3 represents blue spatial spectrum. We train and make use of the simulated “ThreeBodyNet” with 3 strategies. The first strategy is to accept the one of the spatial spectrums respectively and so only one spatial spectrum is computed for the output. Through signature-based localization, the RFID tag can be located if at least one gateway receives its packet. For the second strategy, two of the spatial spectrums are combined into a single input and so three combinations can be obtained (i.e., red and blue, red and green, and green and blue), each of which is fed into “ThreeBodyNet”. Therefore, RFID tag will be located as long as at least two gateways receive the packet by acquiring two spatial spectrums. The third strategy is using all three spatial spectrums by

combining them into a single input by feeding into “ThreeBodyNet” as a whole. Two of the spatial spectrums are dispensable that they minimize the uncertainty from learning randomness. So, RFID tag will be located only in the circumstance that all three gateways acquire its packet simultaneously. Resultantly, input with all spatial spectrums will produce the most accurate result. For the input of “ThreeBodyNet”, RSSI and phase values received from the RFID tag in Scene A Setting 1, a semi-indoor environment with 78.5m² (Table 1), is the raw data for training prediction models of “ThreeBodyNet”, the designated DNN for indoor localization prediction.

```
import torch
import pandas as pd
import torch.nn as nn
from torch.utils.data import random_split, DataLoader, TensorDataset
import torch.nn.functional as F
import numpy as np

path = "Model7.pth"
# Define neural network
class Network(nn.Module):
    def __init__(self, input, output):
        super(Network, self).__init__()

        self.layer1 = nn.Linear(input, 100)
        self.layer2 = nn.Linear(100, 100)
        self.layer3 = nn.Linear(100, 100)
        self.layer4 = nn.Linear(100, 100)
        self.layer5 = nn.Linear(100, 100)
        self.layer6 = nn.Linear(100, 100)
        self.layer7 = nn.Linear(100, output)
```

Figure 15: Part of the code of project.py (1)

```
# Split train, validate and test sets
validate_set = int(len(X) * 0.2)
test_set = int(len(X) * 0.3)
train_set = len(X) - test_set - validate_set
train_set, validate_set, test_set = random_split(data, [train_set, validate_set, test_set])
# Read the data in batches and put into memory
train_loader = DataLoader(train_set, batch_size=16, shuffle=True)
validate_loader = DataLoader(validate_set, batch_size=32)
test_loader = DataLoader(test_set, batch_size=32)

# Define model
model = Network(X.size(1), Y.size(1))
model.to(torch.device("cuda:0" if torch.cuda.is_available() else "cpu"))

# Define loss function and optimizer
loss_function = nn.MSELoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.001, weight_decay=0.0001)
epochs = 200
train(model, epochs, train_loader, validate_loader, optimizer, loss_function)
print('Finish training')
```

Figure 16: Part of the code of project.py (2)

	Gateway 1	Gateway 2	Gateway 3
Prediction 1	✓		
Prediction 2		✓	
Prediction 3			✓
Prediction 4	✓	✓	
Prediction 5	✓		✓
Prediction 6		✓	✓
Prediction 7	✓	✓	✓

Table 3: Gateway(s) used for each prediction

In “ThreeBodyNet”, the neural network consists of 7 layers and each layer consists of 100 neurons for analyzing the input data, RSSI and phase values, from gateways by applying linear transformation to the incoming data in every layer (Figure 15). After that, the model is trained 200 epochs using Adam optimizer with 0.0001 weight decay, 0.001 learning rate, mean squared error (MSE) loss function, training dataset containing 50% of the whole dataset, stored in a CSV file converted from JSON format, and validating dataset containing 20% of the dataset. In each epoch, the overall training loss is the average training loss, measuring the performance of the model with the least value, in the dataset and the overall validating loss is computed by the MSE loss function. The model with the least MSE loss function is saved for prediction of the testing data, containing the remaining 30% of the dataset (Figure 16). Smaller MSE results in smaller Euclidian distance. Here, the purpose of validation dataset, containing 20% of the dataset, and testing dataset, containing 30% of the dataset, are different during the training of the DNN. Validation dataset is to evaluate the performance of the DNN when facing unseen data in order to make sure that it is learning to generalize new data and not only memorizes the training data for continuous improvements. Overfitting is hence prevented using the dataset by finding the optimal balance during training or else the DNN will start simply memorizes the training data instead of learning its generalization. Also, the dataset is to adjust hyperparameters of the DNN such as optimizer, weight decay, and learning rate so that the prediction model with the best performance on validation dataset can be chosen. On the contrary, testing dataset is for the DNN to predict unseen data after training so as to examine the generalization of new data so that its performance in reality is estimated, and rooms of improvement are identified. For instance, if testing dataset contains outliers that preventing the prediction model from performing as expected, manual adjustments are needed. At last, the saved model is loaded for testing and prediction. In the testing, the test loss is computed by the MSE loss function. In the prediction, RSSI, phase values and actual coordinates are displayed and saved in an Excel file as well as the predicted coordinates, the output, for further retrieval in the 3D scatter plot webpage. 7 models

and prediction sets are generated in total. A pair of model and prediction set is generated by RSSI and phase values captured by all 3 gateways. 3 pairs of models and prediction sets are generated by the data captured only by one gateway. Another 3 pairs of models and prediction sets are generated by the data captured by either 2 gateways together (Figure 5 & Table 3).

In addition to “ThreeBodyNet” predicting the coordinates of the RFID tag for indoor localization, a 3D scatter plot, the platform visualizing coordinates in a 3D Cartesian coordinate system by means of webpage, is also developed. The 3D scatter plot now managed to be integrated with the aforementioned DNN for coordinates prediction to show the predicted coordinates computed by the developed prediction model and the corresponding actual coordinates in a GUI given RSSI and phase values. All involved data and coordinates are retrieved from the saved Excel file as the input. User can find out the pairs of actual coordinates and its corresponding predicted coordinates, their pair number, their distance, the list of RSSI and phase values, both of which consist of 16 values for each gateway, of the actual coordinates, and the average RSSI and phase values at a glance. On top of that, users can rotate the 3D scatter plot to spot the distance of the pair of coordinates from a more convenient view (Figure 6).

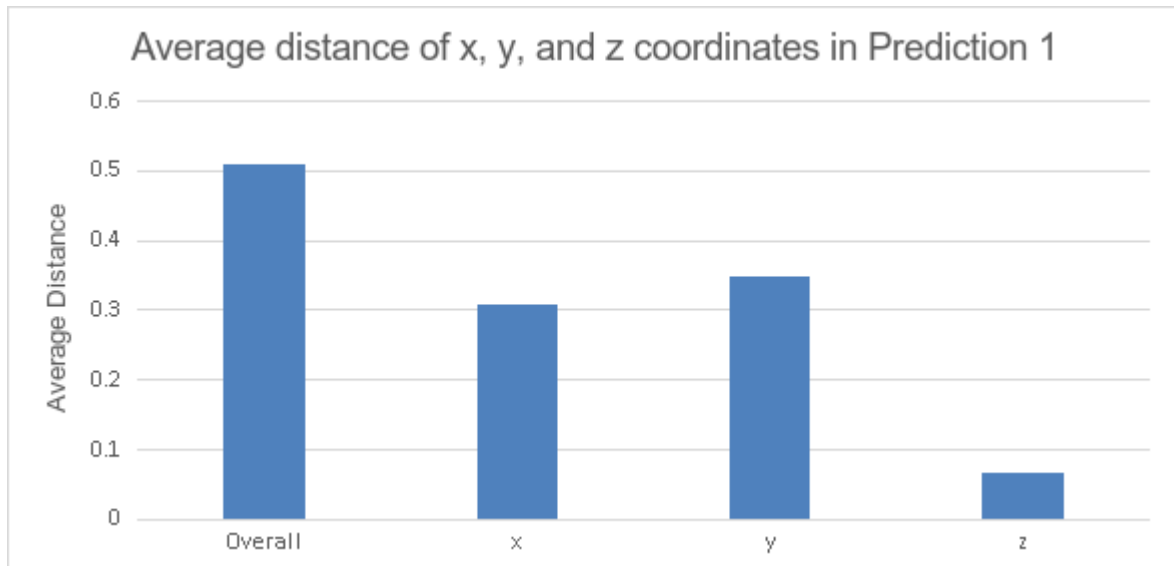


Figure 17: Average distance of overall, x, y, and z coordinates in Prediction 1

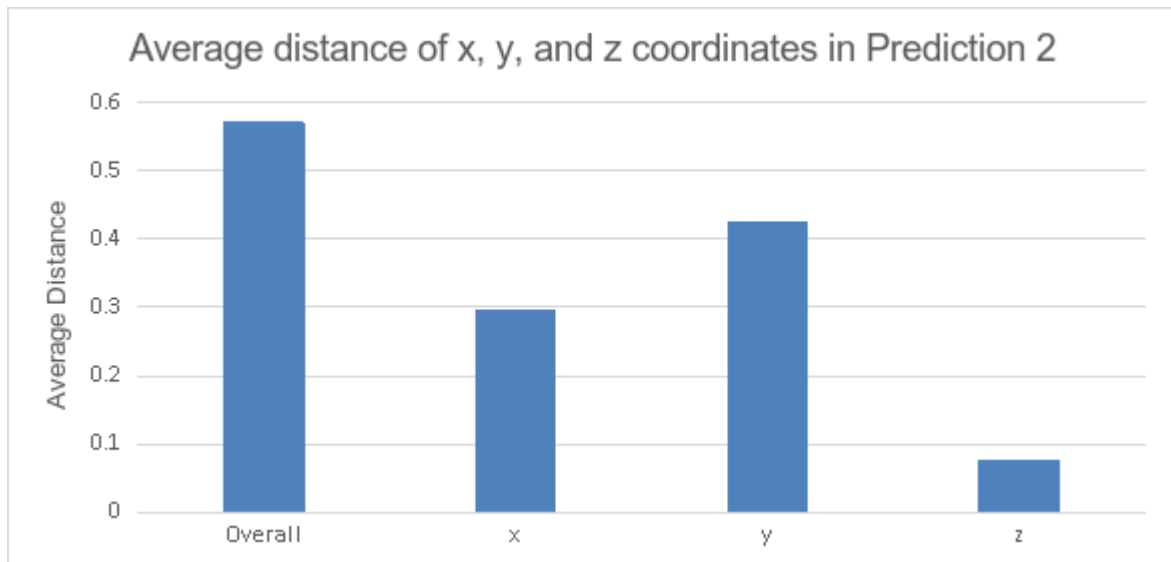


Figure 18: Average distance of overall, x, y, and z coordinates in Prediction 2

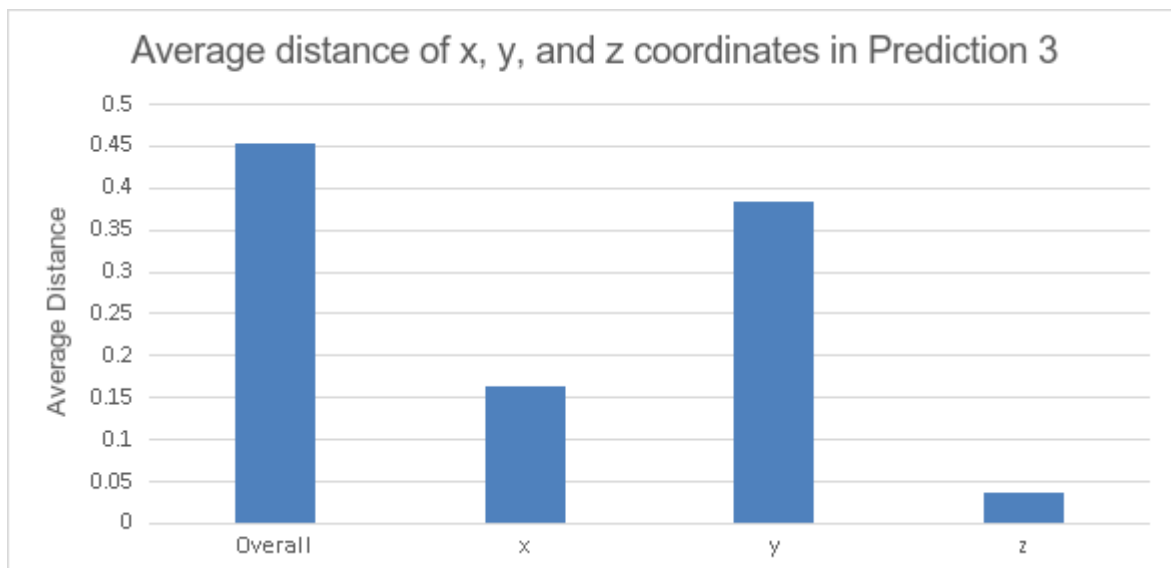


Figure 19: Average distance of overall, x, y, and z coordinates in Prediction 3

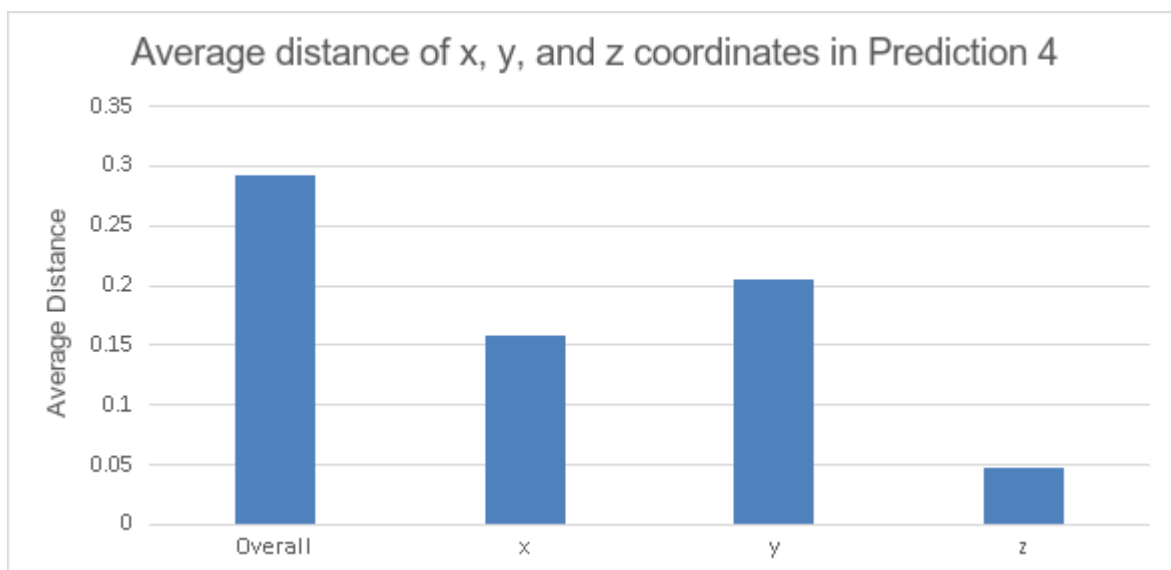


Figure 20: Average distance of overall, x, y, and z coordinates in Prediction 4

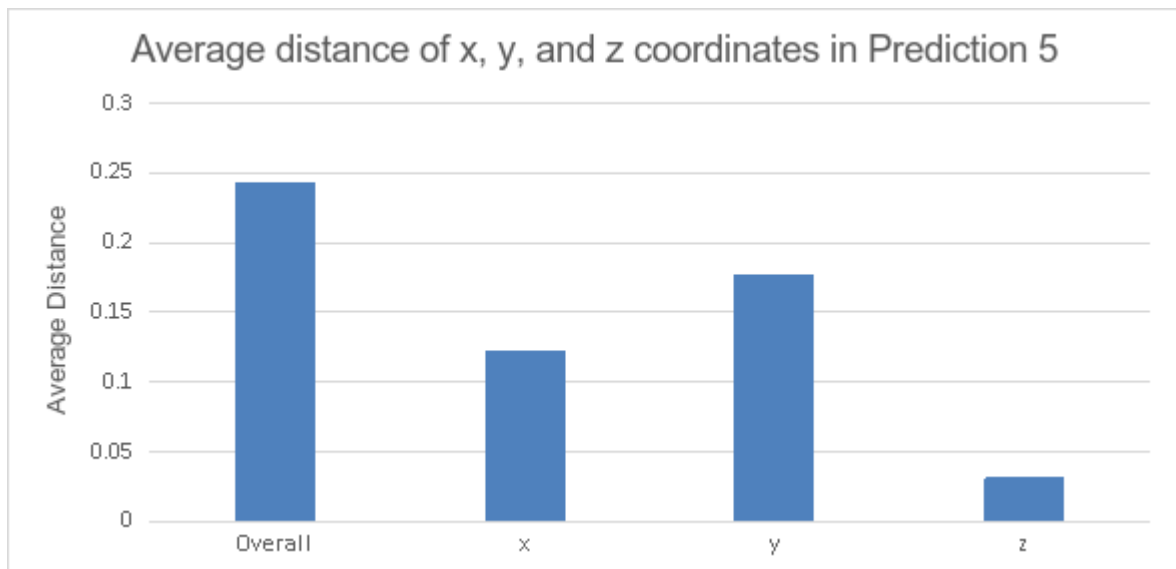


Figure 21: Average distance of overall, x, y, and z coordinates in Prediction 5

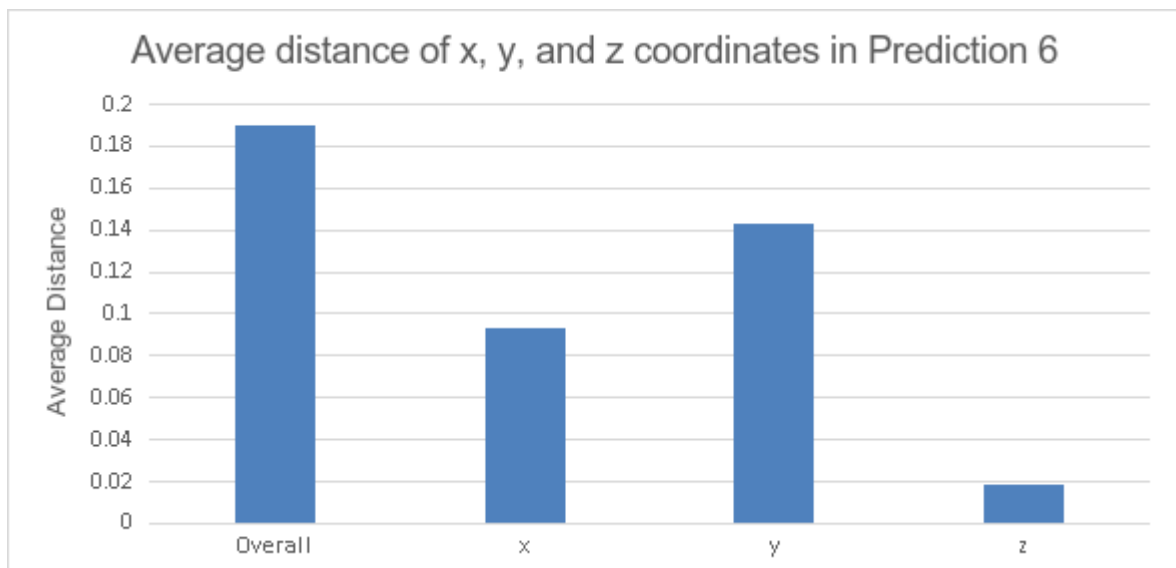


Figure 22: Average distance of overall, x, y, and z coordinates in Prediction 6

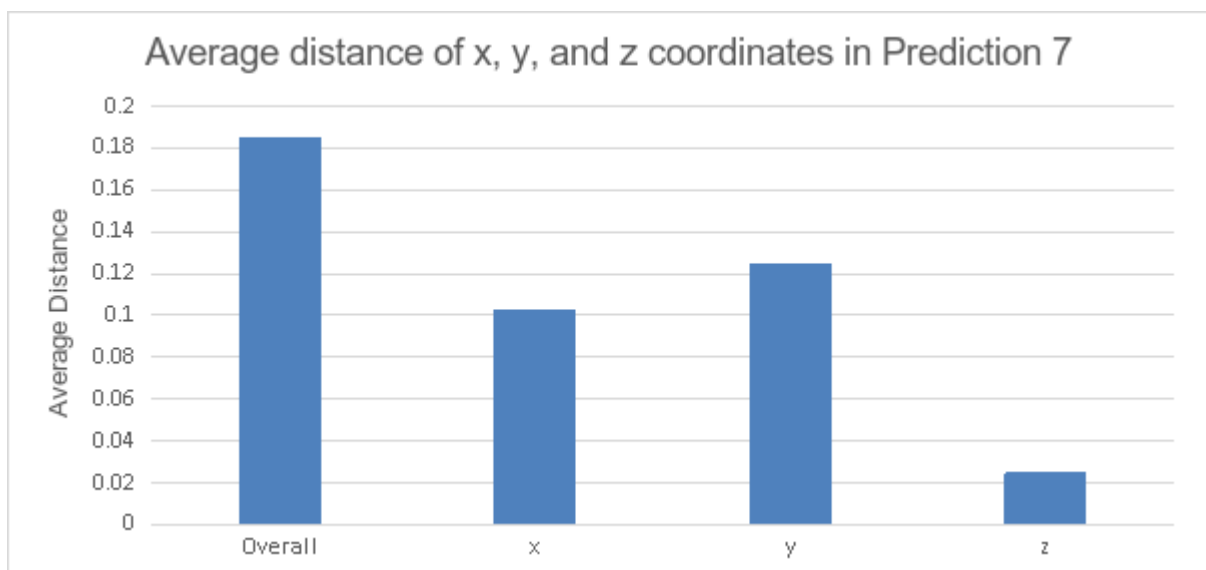


Figure 23: Average distance of overall, x, y, and z coordinates in Prediction 7

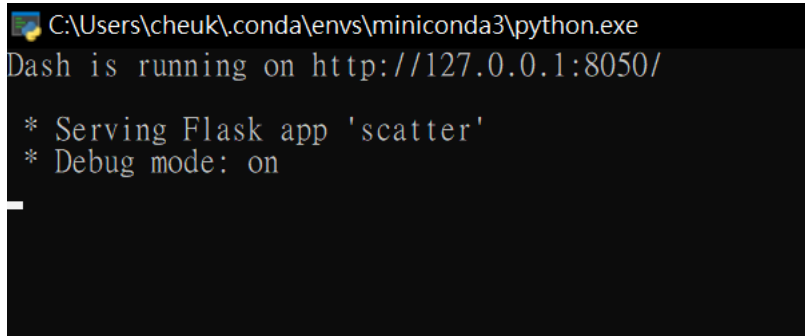
	Median of distance between actual and predicted coordinates (m)	Percentage of number of pairs of actual and predicted coordinates with distance less than 1 meter	Training loss of 200 th epoch	Validation loss of 200 th epoch	Overall test loss	Corresponding prediction model
Prediction 1	0.307	86.9%	0.00537	0.06350	0.09145	Model1.pth
Prediction 2	0.348	85.1%	0.06144	0.16749	0.19302	Model2.pth
Prediction 3	0.270	90.1%	0.00739	0.07595	0.07992	Model3.pth
Prediction 4	0.157	94.5%	0.02349	0.06914	0.09200	Model4.pth
Prediction 5	0.123	96.3%	0.00705	0.03506	0.08477	Model5.pth
Prediction 6	0.107	97.3%	0.00511	0.04753	0.04112	Model6.pth
Prediction 7	0.127	99.1%	0.00423	0.04459	0.03895	Model7.pth

Table 4: Summary of each prediction

After the visualization of coordinates computed the prediction models for indoor localization through a 3D scatter plot, we would like to mathematically represent the holistic performance of each prediction model to discover its extent of errors caused since greater distance between the actual and predicted coordinates indicates more errors and less precision of the prediction model. We find that CDF can address this objective by extracting all distances between each pair of coordinates and sorting them in ascending order. Then, the CDF graphs of each prediction model with x-axis of distance, denoting the distance between the actual and predicted coordinates in the unit of meter and therefore referring to the error of each prediction of coordinates, are generated in PNG format (Figure 7-13). Apart from CDF graphs, the overall average distance and the average distance of x, y, and z coordinates when comparing the actual and predicted coordinates of each prediction model is calculated and visualized by bar chart to find which orientation of coordinate performs the worst and therefore the factor contributing to errors of the indoor localization model is laid bare (Figure 17-23). As a result, improvements against this weakness can be suggested. Also, the median of distance between actual and predicted coordinates and the percentage of number of pairs of actual and predicted coordinates with distance less than 1 meter are counted (Table 4).

4 Implementation and Results

The “project.py” task, the implementation of the aforementioned “ThreeBodyNet”, is structured as the following. First, the DNN with seven layers and each layer consisting of 100 neurons and applying linear transformation are constructed. All involved RSSI and phase values are both the inputs of the first layer of the DNN and its output is the input of the second layer. The whole process continues until reaching the 7th layer, the last layer of the DNN. The processed result is then the final output of the DNN for each epoch of training. In order to make sure that “ThreeBodyNet” performs with least errors in indoor localization prediction, the DNN is to train 200 epochs using Adam optimizer with 0.0001 weight decay, 0.001 learning rate, and 50% of the dataset and validate using 20% of the dataset. In each epoch, the overall training loss is the average training loss of training each item in the dataset by calculating the difference between its predicted coordinates and the actual coordinates for the training dataset and the overall validating loss is computed by the MSE loss function. The model with the least MSE loss function is saved for prediction of the testing data, containing the remaining 30% of the dataset (Figure 16). With small MSE, smaller Euclidian distance is produced. At last, the saved model is loaded for testing and prediction. In the testing, the test loss is computed by the MSE loss function. In the prediction, RSSI, phase values and actual coordinates are displayed and saved in an Excel file as well as the predicted coordinates, the output, for further retrieval in the 3D scatter plot webpage. Each prediction model computed using different sources of raw data, randomly distributed to training dataset, validation dataset, and testing dataset, predicts the coordinates of the RFID tag and saves the results in the Excel file (Figure 5 & Table 3).



```
C:\Users\cheuk\conda\envs\miniconda3\python.exe
Dash is running on http://127.0.0.1:8050/

* Serving Flask app 'scatter'
* Debug mode: on
```

Figure 24: URL of the scatter plot webpage

For the 3D scatter plot, visualizing pairs of actual and predicted coordinates in a webpage of 3D Cartesian coordinate system, is developed in “scatter.py”. After starting the application, it states the URL of the scatter plot webpage, which is “http://127.0.0.1:8050/” (Figure 24). Entering the URL in a browser, user will go to the 3D scatter plot webpage. There, user can find out the relationship between RSSI, phase value, and the distance of actual to predicted coordinates by look at each pair of actual coordinates and its corresponding predicted coordinates, their pair number, their distance, their data consisting of 16 values for each gateway, and the average RSSI and phase values at a glance. Besides, users can rotate the 3D scatter plot to realize the distance of the pair of coordinates from a more convenient view (Figure 6). In the visualization of the indoor localization prediction, we find out that “ThreeBodyNet” managed to compute less error, referring to distance between actual and predicted coordinates, given lower phase values and therefore phase values are inversely proportional to the precision of the prediction by indicating the high quality of RSSI data collected from the RFID tag with low value.

For the CDF computation, “cdf.py” is its implementation that mathematically exemplifying the performance of each prediction model to discover its extent of caused errors, reflected by distance between the actual and predicted coordinates and bringing about the hinderance of the precision of the prediction model. Sorting all distances between each pair of coordinates in ascending order, we produce the CDF graphs of each prediction model in PNG format with x-axis of distance, denoting the distance between the actual and predicted coordinates in the unit of meter and therefore referring to the error of each prediction of coordinates (Figure 7-13).

The CDF graphs are no wonder to exhibit the most inclined slope between the intervals 0m and 1m when it comes to Prediction 7, the prediction taking all three gateways into account and therefore resulting in more minor errors whereas fewer significant errors when predicting coordinates. They exhibit a more inclined slope between the intervals in Prediction 4, Prediction 5, and Prediction 6, all of which makes use of two of the gateways. Since Prediction 1, Prediction 2, and Prediction 3 each only utilizes one gateway, their slopes are the least inclined. Apart from CDF graphs, the overall average distance and the average distance of x, y, and z coordinates when comparing the actual and predicted coordinates of each prediction model is calculated and visualized by bar chart. y-coordinate is discovered to contribute the most to the error of indoor localization prediction ranging from 0.12m to 0.42m and x-coordinate contributes slightly less ranging from 0.09m to 0.3m. However, z-coordinate contributes the least ranging from 0.02m to 0.08m. As a result, the future improvements of the indoor localization prediction are expected to tackle the inaccuracies of the y-coordinate at the first place and then x-coordinate (Figure 17-23). In addition, the median of distance between actual and predicted coordinates and the percentage of number of pairs of actual and predicted coordinates with distance less than 1 meter are counted. One of the predictions only using one gateway reaches the median less than 0.3m and the percentage 90%. The median of predictions using more than one gateway are all less than 0.2m. For the percentage, the prediction making use of all gateways even reaches 99%. Therefore, the number of gateways considered evidently contributes to the accuracy of the indoor localization prediction using “ThreeBodyNet”.

5 Evaluation

“ThreeBodyNet” is implemented by PyTorch framework 1.12.1 and fed with training dataset composed by 50% of the whole dataset, validation dataset composed by 20% of the dataset, and testing dataset composed by 30% of the dataset. All the datasets are from the “Ray” database and randomly distributed. The DNN is built and trained on a laptop with Intel Core i5-10210U processor and 16GB RAM so that its computing power can support a myriad of software and applications needed for the development of the RFID indoor localization prediction system. The DNN is to train 200 epochs using Adam optimizer with 0.0001 weight decay and 0.001 learning rate and takes roughly 10 minutes to train all 200 epochs. However, it only takes 0.1 seconds for each test. Compared with the mentioned first strategy that only considers 1 gateway when making the prediction model, the second strategy considering 2 gateways and the third strategy considering all 3 gateways, the second strategy third strategy improve the accuracy, measured by the mean distance of each pair of actual and predicted coordinates, by 52.7% and 63.7% respectively. Despite significant improvements, the results are not astonishing due to extra computation for additional data from more gateways. Therefore, it is no wonder that the first strategy computes the least satisfied result with more errors generated. On the contrary, in spite of the best performance, the third strategy is hardly practical due to a rigid requirement that the location of the RFID tag must be covered by all three gateways concurrently. However, the first, second, and third strategy are applicable to the area of 100%, 62%, and 20% respectively. Being a trade-off between coverage and precision, the second strategy is prevailed to be the result of “ThreeBodyNet”, the DNN indoor localization prediction. Apart from “ThreeBodyNet”, “iArk”, a current and similar DNN using an antenna array for triangulation, is compared with “ThreeBodyNet”. As expected, “ThreeBodyNet” outperforms than “iArk” in the same setting by separating three body networks and multiple gateways (Figure 4).

Apart from DNN, a few measures tackling the inaccuracies happened when predicting x and y coordinates of the RFID tag are suggested. Spectra generation algorithms from which RSSI and

phase values, the input of the DNN, are converted in the representation of images are the starting point to improve the accuracy. Bartlett and MVDR are the suggested algorithms minimizing errors during prediction most effectively. For Bartlett algorithm, it tests the assumption of equal variances of RSSI across different groups, the scenarios that the data are collected. Similar variances result in a valid dataset for prediction of coordinates by means of DNN. For MVDR algorithm, it commonly used in signal processing to increase the signal-to-noise ratio (SNR) for desired sets of RSSI and phase values. Capturing RSSI and phase values from the RFID tag by array of antennas, the algorithm then weights them to steer the array for the estimation of the direction of arrival (DOA) of the desired signal by suppressing the interference and noise from other directions. Afterwards, the angle of arrival (AOA) of the signal, combining from multiple antenna arrays, is computed to predict the location of the RFID tag by triangulation. Eventually, the accuracy of the algorithm can be measured by MSE. These two algorithms exhibit smaller errors in x and y coordinates of within 0.1m (Han, 2011). So, these conventional algorithms are able to achieve the highest accuracy unexpectedly on account of preserving more original information, being beneficial for the DNN to identify the pattern of each set of data possibly.

In addition, phase estimation algorithms for preprocessing phase values in the combination of low noise phase estimator (LNPE), Kalman Filter (KF), and cyclic prefix and cyclic suffix (CC) is introduced to the training of “ThreeBodyNet” for higher accuracy of prediction. LNPE estimates the relative timing and position of the signal with respect to the reference signal by either finding the phase maximizing the likelihood of the signal to be computed or analyzing the frequency of the signal by Fourier Transform (FT). KF predicts the state of the system and updates based on the previous measurements by statical methods overcoming noises and various uncertainties and therefore preforms well when it comes to noisy and incomplete measurements. CC helps on condition that the direction of the device is near to the margins of the spatial spectrum by mitigating inter-symbol interference (ISI) due to multipath propagation. Inserting a guard interval, CC creates a cyclic extension transmitted along with the signal and discard after

being received so that the phase values, representing the remaining parts of the signal, can be prevented from ISI and so the reliability of RSSI is maintained for training the DNN (Grant, 2010). As has been shown, suggestions tackling errors of the indoor localization prediction starting from RSSI and phase values respectively are listed.

There are several discussions of this research are put forward. First, the study of the interference is yet to be found out by not clarifying the extent of errors brought about by metal and human interference. The study is expected whereas not carried out due to limited amount of data required for enough training dataset to produce a reliable model with the related scenarios, dynamic and constantly changing environments (Holland, 2009). Also, the interference can be studied by intentionally placing obstacles, people, or even antennas in front of the RFID tag so that investigations of how different sources of interference affect the accuracy of the indoor localization prediction can be carried out (Zeng, 2019). Besides, if the accuracy of the indoor localization prediction is not restricted within 0.3m or there was few RF interference, alternative methods such as linear regression, random forest, and decision tree may also be attempted by virtue of decreasing computation cost.

6 Conclusion

The “Ray” database, consisting of a general-purpose, protocol-free, and million-scale dataset with millimeter-level labels, is adopted for training “ThreeBodyNet”, the deep neural network dedicated for indoor localization prediction. Consisting of raw data in RSSI and phase values of the RFID tag collected from a myriad of settings and scenes, the database helps further understanding of indoor localization prediction by DNN with high quality of raw data in order that prediction models computed can be utilized in practical scenarios such as searching for a particular item out of numerous items in an indoor environment by means of RFID for localization prediction.

7 Resources Utilized

In order to build the RFID indoor localization system, a laptop with Intel Core i5-10210U processor and 16GB RAM is needed so that the computing power of the laptop can support a myriad of software and applications needed for building the RFID indoor localization system. For file format of data storage, Microsoft Excel is the spreadsheet software storing the given RFID data, the actual position of RFID tags, and the position of RFID tags predicted by several neural network models for the localization system. Visual Studio, providing an integrated development environment (IDE) for Python, is the platform for development and feature implementation of the indoor localization system.

8 References

- ACM MobiCom 2022 (2022). *Deep Learning on Indoor Localization: A Million-Scale Database with Millimeter-Level Labels*
- DataFlair (n.d.). *Deep Learning vs Machine Learning – Demystified in Simple Words*. Retrieved from <https://data-flair.training/blogs/deep-learning-vs-machine-learning/>
- Fox et al. (2004). *Mapping and Localization with RFID Technology*
- Grant et al. (2010). *Accurate Localization of RFID Tags Using Phase Difference*
- Han et al. (2011). *Improving Accuracy for 3D RFID Localization*
- Holland, W. (2009). *Development of an Indoor Real-time Localization System Using Passive RFID Tags and Artificial Neural Network*
- Jin et al. (2006). *An Indoor Localization Mechanism Using Active RFID Tag*
- Shen et al. (2019). *PRDL: Relative Localization Method of RFID Tags via Phase and RSSI Based on Deep Learning*
- SmartX (n.d.). *What is the difference between RFID and a Beacon?* Retrieved from <https://smartrxhub.com/what-is-the-difference-between-rfid-and-a-beacon/>
- Wolfewicz, A. (2022). *Deep Learning vs. Machine Learning – What’s The Difference?* Retrieved from <https://levity.ai/blog/difference-machine-learning-deep-learning>
- Zeng et al. (2019). *UHF RFID Indoor Positioning System With Phase Interference Model Based on Double Tag Array*